Namelist options in FV^3

Lucas Harris
and the GFDL FV^3 Team

6 March 2017

FV^3 Documentation

Proposed NCEP Office Note 472 (?) for distribution with public code release
Draft in limited distribution

Comprehensive document describing solver algorithm, configuration, and usage

Focusing on solver and diffusion options in this presentation; see document for complete coverage

FV^3: THE GFDL FINITE-VOLUME CUBED-SPHERE DYNAMICAL CORE

The GFDL FV^3 Team

March 5, 2017

NCEP Office Note 472 (Proposed) **(DRAFT, unfinished)**
Not for distribution beyond
GFDL, GSFC, EMC, CAPS, PSD, AOML
Contact GFDL FV^3 support
(oar.gfdl.fvGFS_support@noaa.gov) if found elsewhere

1

FV^3 namelist options

```
&fv_core_nml
        npx   = 769
        npy   = 769
        npz   = 63
        n_sponge = 8
        tau = 5.
        rf_cutoff = 8.e2
        d2_bg_k1 = 0.16
        d2_bg_k2 = 0.02
        hydrostatic = .F.
        k_split  = 2
        n_split  = 6
        fv_sg_adj = 1800
        nord =  2
        d4_bg = 0.15
        vtdm4 = 0.
        do_vort_damp
= .false.
        d_con = 0.
        hord_mt =  8
        hord_vt =  8
        hord_tm =  8
        hord_dp =  8
        hord_tr =  8
```

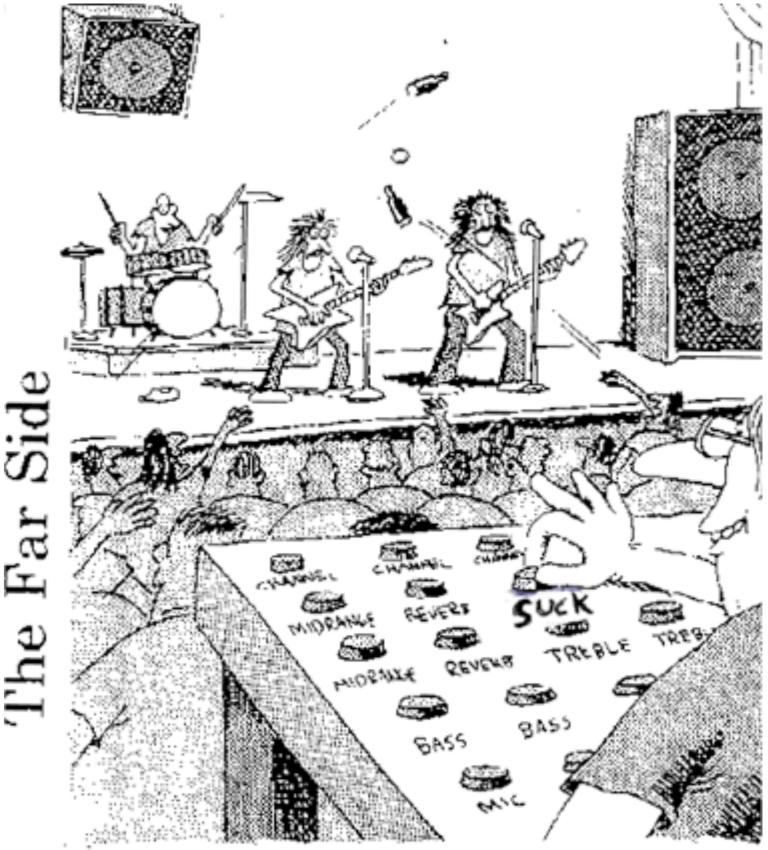
Describing key parameters in a sample configuration used for 13-km forecasts, configured for Zhao-Carr microphysics

Some parameters (hord_xx) will be updated for public release NEMS and CM4 versions; changes in behavior will be noted below

WARNING

If you don’t know what an option does, don’t mess with it

Unfortunately no do_what_I_want = .true. option



Raymond's last day as the band's sound technician.

Domain specification

```
&fv_core_nml
        npx   = 769
        npy   = 769
        npz   = 63
        n_sponge = 8
        tau = 5.
        rf_cutoff = 8.e2
        d2_bg_k1 = 0.16
        d2_bg_k2 = 0.02
        hydrostatic = .F.
        k_split  = 2
        n_split  = 6
        fv_sg_adj = 1800
        nord =  2
        d4_bg = 0.15
        vtdm4 = 0.
        do_vort_damp
= .false.
        d_con = 0.
        hord_mt =  8
        hord_vt =  8
        hord_tm =  8
        hord_dp =  8
        hord_tr =  8
```

npx and npy control the number of grid corners across a cube face; subtract one to get the number of grid cells

c768 corresponds roughly to ⅛ degree, or 12 km global average grid-cell width

On the global grid npx = npy
But they can differ on a nest

npz is the number of grid levels, with a hard-coded specification of level placement

64-level model top at ~0.6 mb

Good configurations are nontrivial to design

Domain specification

```
&fv_core_nml
        npx   = 769
        npy   = 769
        npz   = 63
        n_sponge = 8
        tau = 5.
        rf_cutoff = 8.e2
        d2_bg_k1 = 0.16
        d2_bg_k2 = 0.02
        hydrostatic = .F.
        k_split  = 2
        n_split  = 6
        fv_sg_adj = 1800
        nord =  2
        d4_bg = 0.15
        vtdm4 = 0.
        do_vort_damp
= .false.
        d_con = 0.
        hord_mt =  8
        hord_vt =  8
        hord_tm =  8
        hord_dp =  8
        hord_tr =  8
```

npz is the number of grid levels, with a hard-coded specification of level placement

63-level model top at constant 0.6 mb pressure

Good configurations are nontrivial to design

- Enhanced resolution in PBL (and possibly UTLS)
- Smooth variation in pressure levels
- Choose a good model top

Timestepping

```
&fv_core_nml
        npx          = 769
        npy          = 769
        npz          = 63
        n_sponge = 8
        tau = 5.
        rf_cutoff = 8.e2
        d2_bg_k1 = 0.16
        d2_bg_k2 = 0.02
        hydrostatic = .F.
        k_split  = 2
        n_split  = 6
        fv_sg_adj = 1800
        nord =  2
        d4_bg = 0.15
        vtdm4 = 0.
        do_vort_damp = .false.
        d_con = 0.
        hord_mt =  8
        hord_vt =  8
        hord_tm =  8
        hord_dp =  8
        hord_tr =  8
```

```
&coupler_nml
        dt_atmos = 225
```

dt_atmos is the physics timestep: **225 s**, matching GFS

Physics is applied forward-in-time, consistent with

FV[3] dynamics

Vertical remapping is done k_split times per physics timestep

112.5 s: Lagrangian vertical coordinate has *no* Courant number restriction!

k_split > 1 can enhance stability for the same acoustic timestep, with a minimal performance degradation

Timestepping

```
&fv_core_nml
        npx           = 769
        npy           = 769
        npz           = 63
        n_sponge = 8
        tau = 5.
        rf_cutoff = 8.e2
        d2_bg_k1 = 0.16
        d2_bg_k2 = 0.02
        hydrostatic = .F.
        k_split  = 2
        n_split  = 6
        fv_sg_adj = 1800
        nord =  2
        d4_bg = 0.15
        vtdm4 = 0.
        do_vort_damp = .false.
        d_con = 0.
        hord_mt =  8
        hord_vt =  8
        hord_tm =  8
        hord_dp =  8
        hord_tr =  8
```

```
&coupler_nml
        dt_atmos = 225
```

Acoustic solver and horizontal dynamics called n_split times between vertical remappings

Dynamics advanced by forward-backward timestepping, with sub-cycled tracer advection

Acoustic timestep = dt_atmos / k_split * n_split

18.75 s in this example

Monotonic scheme

```
&fv_core_nml
        npx          = 769
        npy          = 769
        npz          = 63
        n_sponge = 8
        tau = 5.
        rf_cutoff = 8.e2
        d2_bg_k1 = 0.16
        d2_bg_k2 = 0.02
        hydrostatic = .F.
        k_split  = 2
        n_split  = 6
        fv_sg_adj = 1800
        nord =  2
        d4_bg = 0.15
        vtdm4 = 0.
        do_vort_damp = .false.
        d_con = 0.
        hord_mt =  8
        hord_vt =  8
        hord_tm =  8
        hord_dp =  8
        hord_tr =  8
```

Non-monotonic ("linear") scheme

```
&fv_core_nml
        npx          = 769
        npy          = 769
        npz          = 63
        n_sponge = 8
        tau = 5.
        rf_cutoff = 8.e2
        d2_bg_k1 = 0.16
        d2_bg_k2 = 0.02
        hydrostatic = .F.
        k_split  = 2
        n_split  = 6
        fv_sg_adj = 1800
        nord =  2
        d4_bg = 0.15
        vtdm4 = 0.04
        do_vort_damp = .true.
        d_con = 1.
        hord_mt =  5
        hord_vt =  5
        hord_tm =  5
        hord_dp =  5
        hord_tr =  8
```

Optimized monotonic and non-monotonic ("linear") schemes for computing fluxes. Tracer advection is *always* monotonic (8; -8 in NGGPS code) and is *never* explicitly diffused

Monotonic scheme (8; -8 in NGGPS code) is intrinsically diffusive to 2-delta-waves. Explicit horizontal damping from 6th-order (nord = 2) divergence damping

No explicit ("vorticity") damping on other fluxes

do_vort_damp = .false.

Monotonic scheme

```
&fv_core_nml
    npx          = 769
    npy          = 769
    npz          = 63
    n_sponge = 8
    tau = 5.
    rf_cutoff = 8.e2
    d2_bg_k1 = 0.16
    d2_bg_k2 = 0.02
    hydrostatic = .F.
    k_split  = 2
    n_split  = 6
    fv_sg_adj = 1800
    nord =  2
    d4_bg = 0.15
    vtdm4 = 0.
    do_vort_damp = .false.
    d_con = 0.
    hord_mt =  8
    hord_vt =  8
    hord_tm =  8
    hord_dp =  8
    hord_tr =  8
```

Non-monotonic (“linear”) scheme

```
&fv_core_nml
    npx          = 769
    npy          = 769
    npz          = 63
    n_sponge = 8
    tau = 5.
    rf_cutoff = 8.e2
    d2_bg_k1 = 0.16
    d2_bg_k2 = 0.02
    hydrostatic = .F.
    k_split  = 2
    n_split  = 6
    fv_sg_adj = 1800
    nord =  2
    d4_bg = 0.15
    vtdm4 = 0.04
    do_vort_damp = .true.
    d_con = 1.
    hord_mt =  5
    hord_vt =  5
    hord_tm =  5
    hord_dp =  5
    hord_tr =  8
```

Non-monotonic scheme (5; 6/-5 in NGGPS code) applies *no* monotonicity constraint (“linear”, “unlimited”), only a 2dx filter to suppress oscillations.

Needs consistent damping to vorticity and other fluxes. This damping (vtdm4) should be weaker than the divergence damping.

Artificial diffusion

```
&fv_core_nml
        npx   = 769
        npy   = 769
        npz   = 63
        n_sponge = 8
        tau = 5.
        rf_cutoff = 8.e2
        d2_bg_k1 = 0.16
        d2_bg_k2 = 0.02
        hydrostatic = .F.
        k_split  = 2
        n_split  = 6
        fv_sg_adj = 1800
        nord =  2
        d4_bg = 0.15
        vtdm4 = 0.04
        do_vort_damp
= .true.
        d_con = 1.
        hord_mt =  5
        hord_vt =  5
        hord_tm =  5
        hord_dp =  5
        hord_tr =  8
```

Explicit divergence damping necessary since there is **no** implicit diffusion to divergence

Strength d4_bg should range from 0.10 to 0.16

All damping applied **along** Lagrangian surfaces

Optional damping to other fluxes (vorticity, air mass, w, θ) controlled by vtdm4 “vorticity damping”

Strongly recommended if non-mono scheme is used

Vortical and divergent modes damped separately. vtdm4 should be much smaller than d4_bg

Artificial diffusion

```
&fv_core_nml
        npx   = 769
        npy   = 769
        npz   = 63
        n_sponge = 8
        tau = 5.
        rf_cutoff = 8.e2
        d2_bg_k1 = 0.16
        d2_bg_k2 = 0.02
        hydrostatic = .F.
        k_split  = 2
        n_split  = 6
        fv_sg_adj = 1800
        nord =  2
        d4_bg = 0.15
        vtdm4 = 0.04
        do_vort_damp
= .true.
        d_con = 1.
        hord_mt =  5
        hord_vt =  5
        hord_tm =  5
        hord_dp =  5
        hord_tr =  8
```

The local dissipated kinetic energy from flux damping can be added back as heat (d_con = 1) for better conservation of lost energy

Can cause instability if vtdm4 is small

Set d_con to 0 if vtdm4 < 0.02

Damping order is 2 x (nord+1); fourth, sixth, and eighth-order scale-selective damping are available

Rayleigh damping and sponge layer

```
&fv_core_nml
    npx          = 769
    npy          = 769
    npz          = 63
    n_sponge = 8
    tau = 5.
    rf_cutoff = 8.e2
    d2_bg_k1 = 0.16
    d2_bg_k2 = 0.02
    hydrostatic = .F.
    k_split  = 2
    n_split  = 6
    fv_sg_adj = 1800
    nord =  2
    d4_bg = 0.15
    vtdm4 = 0.
    do_vort_damp = .false.
    d_con = 0.
    hord_mt =  8
    hord_vt =  8
    hord_tm =  8
    hord_dp =  8
    hord_tr =  8
```

Rayleigh damping is applied *consistently* to (u, v, w) with timescale tau (here 5 days)

Lost kinetic energy converted to heat

Rayleigh damping is only applied above rf_cutoff (in Pa); the top 6 layers in this case

Should be tuned with GWD to produce the most stable and noise-free result

Consider more sponge layers with weaker damping (larger tau)

Rayleigh damping and sponge layer

```
&fv_core_nml
        npx          = 769
        npy          = 769
        npz          = 63
        n_sponge = 8
        tau = 5.
        rf_cutoff = 8.e2
        d2_bg_k1 = 0.16
        d2_bg_k2 = 0.02
        hydrostatic = .F.
        k_split  = 2
        n_split  = 6
        fv_sg_adj = 1800
        nord =  2
        d4_bg = 0.15
        vtdm4 = 0.
        do_vort_damp = .false.
        d_con = 0.
        hord_mt =  8
        hord_vt =  8
        hord_tm =  8
        hord_dp =  8
        hord_tr =  8
```

Sponge layer is active in the top two layers of the model, using second-order horizontal damping to suppress wave reflection

> d2_bg_k1 should be between 0.16 and 0.2
> d2_bg_k2 should be much smaller

A 2dz filter controls local dynamic instability in top n_sponge layers only

> Relaxes Ri < 1 nonlinear instabilities with timescale fv_sg_adj

Diffusion and damping

Well-configured numerical diffusion, damping, sponge layers, and GWD can greatly improve the stability of the model.

Decreasing the timestep should be a last resort:
consider re-tuning diffusion first

Keeping forecast skill and quality in mind, of course

Damping and timestep length are physics-dependent. Different drag schemes and prognostic microphysics may require different damping and timesteps.
Physics and dynamics need to be optimized together.

Other options of interest

kord_{tm,mt,wz,tr} control cubic-spline vertical remapping scheme.
kord_tm < 0 remaps T, which is much more accurate than remapping θ

> kord_xx = 9 is monotonic, while 10 is non-monotonic with 2dz filter on spline.
> 11 is non-monotonic with no filter

dddmp is the coefficient for 2D nonlinear Smagorinsky damping, which is more flow-dependent than linear damping. Values of 0.1 or 0.2 are recommended.

nwat, dnats, and z_tracer will be very useful when implementing advanced microphysics; ask us for advice

Other options of interest

consv_te controls amount of energy lost by solver which is restored by energy fixer (global grid only). AM4 uses 0.6 to reduce imbalance to < 0.01 in AMIP runs.

print_freq controls stdout diagnostics:
frequency (hr) if > 0; period (# of dt_atmos) if < 0

range_warn, fv_debug, and no_dycore are very useful debugging tools controlling checking of invalid values, printing out many more diagnostics, and running the model in column-physics mode

Stretched grid configuration

```
&fv_core_nml
        npx         = 769
        npy         = 769
        npz         = 63
  ....
        nord =  1
        d4_bg = 0.12
        ...
  do_schmidt = .true.
  target_lat = 35.5
  target_lon = -97.5
  stretch_fac = 3.0
```

Grid stretching allows simple, easy local grid refinement within a single global grid.

Enable stretching with do_schmidt = .T.
Set region center with target_lat and target_lon

Refinement factor given by stretch_fac > 1.
Larger values give a smaller high-res region.
Remember to reduce timestep when stretching!

Use fourth-order damping (nord = 1) for stretched grid

Nested grid configuration

```
&fv_core_nml ! nested grid
   npx     = 1729
   npy     = 1441
   ntiles  = 1
   npz     = 63
   k_split = 4
   n_split = 5
```

```
&nest_nml
   ngrids = 2
   nest_pes = $npes_g1,$npes_g2
   p_split = 1
/
```

```
&fv_core_nml   ! coarse grid
          npx          = 769
          npy          = 769
          npz          = 63
   ....
      do_schmidt = .true.
   target_lat = 35.5
   target_lon = -97.5
   stretch_fac = 1.5
```

Each grid gets a separate, complete namelist file input.nml, input_nest02.nml

Physics and infrastructure can be configured separately on each grid

Showing example from c768r15n3---3 km over CONUS---with GFDL MP

dt_atmos = 90: Δt = 4.5 sec

Nested grid configuration

```
&fv_core_nml ! nested grid
    npx      = 1729
    npy      = 1441
    ntiles   = 1
    npz      = 63
    k_split  = 4
    n_split  = 5
```

```
&nest_nml
   ngrids = 2
   nest_pes = $npes_g1,$npes_g2
   p_split = 1
/
```

```
&fv_core_nml   ! coarse grid
        npx            = 769
        npy            = 769
        npz            = 63
    ....
       do_schmidt = .true.
    target_lat = 35.5
    target_lon = -97.5
    stretch_fac = 1.5
```

Both namelist files need a nest_nml to specify ngrids (currently limited to 2), processors for each grid, and number of BC/two-way updates per physics timestep (p_split; +1 recommended)

Rotate coarse grid to center tile over nested grid, and stretch as desired

Currently both grids need to have same npz (working on "remap BCs" to support differing vertical levels)

Nested grid configuration

```
&fv_core_nml ! nested grid
   npx      = 1729
   npy      = 1441
   ntiles   = 1
   npz      = 63
   k_split  = 4
   n_split  = 5
   nord =  3
   dddmp = 0.1
   d4_bg = 0.08
   vtdm4 = 0.005
   do_vort_damp = .T.
   d_con = 0.0
   ...
   nested = .true.
   twowaynest = .true.
   parent_grid_num = 1
   parent_tile = 6
   refinement = 3
   ioffset = 97
   joffset = 165
   nestupdate = 7
```

Several new options must be added to enable nested grid. Only npx, npy, ioffset, joffset are widely configurable, and must match values given for initial conditions.

ioffset, joffset control location of first refined coarse grid cell. This is derivable from preproc tool configuration. (Work is being done to simplify nested-grid setup)

Nested grid configuration

```
&fv_core_nml ! nested grid
    npx      = 1729
    npy      = 1441
    ntiles   = 1
    npz      = 63
    k_split  = 4
    n_split  = 5
    nord =  3
    dddmp = 0.1
    d4_bg = 0.08
    vtdm4 = 0.005
    do_vort_damp = .T.
    d_con = 0.0
    ...
    nested = .true.
    twowaynest = .true.
    parent_grid_num = 1
    parent_tile = 6
    refinement = 3
    ioffset = 97
    joffset = 165
    nestupdate = 7
```

Damping can be greatly reduced on a limited domain (No Himalayas! No Andes!)

Here using 8th order damping (nord = 3) and much reduced divergence and flux damping. d_con has been disabled.

Also using Smagorinsky-like nonlinear horizontal diffusion (dddmp = 0.1)

Zhao-Carr MP will probably need greater diffusion (nord = 2) than shown here